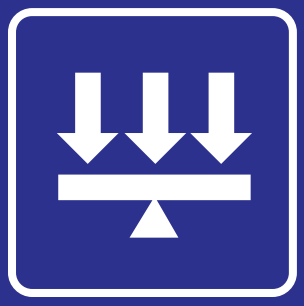


FORTINET

FortiOS™ Handbook - Server Load Balancing

VERSION 6.0.0

FORTINET DOCUMENT LIBRARY

http://docs.fortinet.com

FORTINET VIDEO GUIDE

http://video.fortinet.com

FORTINET BLOG

https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT

https://support.fortinet.com

http://cookbook.fortinet.com/how-to-work-with-fortinet-support/

FORTIGATE COOKBOOK

http://cookbook.fortinet.com

FORTINET TRAINING SERVICES

http://www.fortinet.com/training

FORTIGUARD CENTER

http://www.fortiguard.com

FORTICAST

http://forticast.fortinet.com

END USER LICENSE AGREEMENT

http://www.fortinet.com/doc/legal/EULA.pdf

FORTINET PRIVACY POLICY

https://www.fortinet.com/corporate/about-us/privacy.html

FEEDBACK

Email: techdocs@fortinet.com

March 29, 2018

FortiOS™ Handbook - Server Load Balancing

01-600-480929-20180329

TABLE OF CONTENTS

Change Log

Date	Change Description
March 29, 2018	Initial release.

Introduction

FortiOS server load balancing includes the features you would expect of any server load balancing solution. Traffic can be distributed across multiple backend servers based on multiple methods including static (failover), round robin, weighted to account for different sized servers, or based on the health and performance of the server including round trip time, number of connections. The load balancer supports HTTP, HTTPS, IMAPS, POP3S, SMTPS, SSL or generic TCP/UDP or IP protocols. Session persistence is supported based on the SSL session ID or based on an injected HTTP cookie.

Before you begin

Before you can configure server load balancing on the GUI go to **System > Feature Visibility** and turn on **Load Balance**. Its in the **Additional Features** list.

To be able to use all of the features described in this chapter you should go to **System > Settings** and setting the **Inspection Mode** to **Proxy**. If **Inspection mode** is set to **Flow-based**, you can only configure Virtual Servers with Type set to HTTP, TCP, UDP, or IP. Proxy mode is required for persistence, HTTP Multiplexing, SSL offloading and other advanced HTTP and SSL features.

How this chapter is organized

This document contains detailed information about how to configure FortiOS server load balancing to load balance various types of traffic to multiple backend servers. This document describes all server load balancing configuration options and contains detailed configuration examples.

This FortiOS Handbook chapter contains the following sections:

Inside FortiOS: Server Load Balancing highlights the features and benefits of FortiOS server load balancing.

Basic load balancing configuration example introduces FortiOS server load balancing by providing a basic configuration example.

Configuring load balancing describes how to configure FortiOS server load balancing.

HTTP and HTTPS load balancing, multiplexing, and persistence describes FortiOS server load balancing features that support load balancing HTTP and HTTPS sessions.

SSL/TLS load balancingdescribes FortiOS server load balancing features that support load balancing SSL and TLS sessions.

IP, TCP, and UDP load balancing describes FortiOS server load balancing features that support load balancing IP, TCP, and UDP sessions.

The following configuration examples are also included:

- Example HTTP load balancing to three real web servers
- Example Basic IP load balancing configuration
- Example Adding a server load balance port forwarding virtual IP

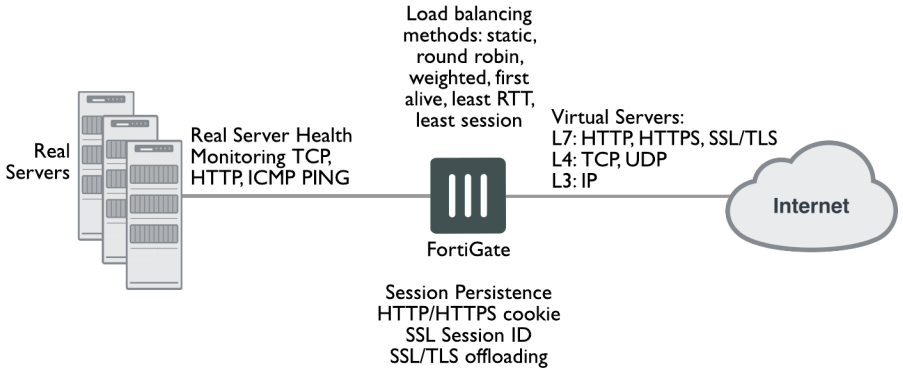
- Example Weighted load balancing configuration
- Example HTTP and HTTPS persistence configuration

Inside FortiOS: Server Load Balancing

Server load balancing distributes workloads across multiple network servers, allowing simultaneous IPv4, IPv6, IPv4 to IPv6 and IPv6 to IPv4 requests to be handled quickly and reliably.

Server Load Balancing combined with NGFW and UTM protection

By introducing comprehensive server load balancing functionality to Next Generation Firewall (NGFW) and Unified Threat Management (UTM) solutions FortiOS takes threat protection to a whole new level. Rather than going to the expense of deploying multiple solutions to protect your server farm, you can combine firewalling, NGFW, UTM and load balancing into a single FortiGate unit or cluster. The benefit of consolidation is not only limited to cost.



Key Features & Benefits

Increased resilience	A consolidated solution results in significantly simplified network architecture. High availability can be provided for all technologies with just a pair of devices rather than several.
Reduced operational overheads	A unified management solution consisting of a single GUI, logging and reporting, SNMP monitoring and other management functions will significantly reduce the resources required to manage the multiple technology areas. A consolidated solution provides a single point of contact for support and renewals rather than having to deal with multiple vendors.

The FortiOS server load balancing feature set contains all of the features you would expect of a server load balancing solution. Traffic can be balanced across multiple backend servers based on multiple load balancing schedules including static (failover), round robin, weighted to account for different sized servers, or based on the health and performance of the server including round trip time and number of connections.

The load balancer supports HTTP, HTTPS, IMAPS, POP3S, SMTPS, SSL/TLS, and generic TCP/UDP and IP protocols. Session persistence is supported based on the SSL session ID, based on an injected HTTP cookie, or based on the HTTP or HTTPS host. SSL/TLS load balancing includes protection from protocol downgrade

attacks. Server load balancing is supported on most FortiGate devices and includes up to 10,000 virtual servers on our high end systems.

SSL/TLS offloading

With more and more critical business applications being made available online and in the cloud, the demand for secure remote continues to increase. While securing web and email applications with SSL/TLS is essential, this protection adds significant performance overheads. An SSL/TLS protected application running on a standard server will perform all the costly encryption/decryption and key exchange routines in software which uses vital CPU resources that should be available for running the application. The consequence of this is that many more or more powerful servers are required to deliver the application.

FortiGate SSL/TLS offloading is designed with the explosion of SSL/TLS applications in mind. The key exchange and encryption/decryption tasks are offloaded to the FortiGate unit where they are accelerated using FortiASIC technology providing significantly more performance than a standard server or load balancer could handle. This frees up valuable resources on the server farm which can be used to run a more responsive business. Server load balancing offloads most SSL/TLS versions including SSL 3.0, TLS 1.0 and TLS 1.2 and supports full mode or half mode SSL offloading with DH key sizes up to 4096 bits.

SSL/TLS content inspection

Traditionally, SSL encrypted application data would be invisible to any border gateway filtering solution. This is because the encryption process prevents the payload of any connection from being seen other than by the communicating systems. FortiGate SSL Offloading allows the application payload to be inspected before it reaches your servers; preventing intrusion attempts, blocking viruses, stopping unwanted applications, and preventing data leakage. SSL/TLS content inspection supports TLS versions 1.0, 1.1, and 1.2 and SSL versions 1.0, 1.1, 1.2, and 3.0.

Health Check

Health checking can be enabled to prevent load balancing traffic from being sent to a non-functioning real server. Real server health can be monitored using ICMP ping or more sophisticated TCP testing. The most comprehensive test is HTTP which verifies that the HTTP application is responding and that it is returning the correct content.

Health checking removes real servers from the load balancing cluster which are returning invalid content. The removal of real servers from the clusters is based on the Interval, Timeout and Retry Settings:

Interval	How often to test the server.
Timeout	What maximum response time is permissible before a server is treated as non-functional.
Retry	How many failures before the server is considered “dead” and removed from the cluster.

Server Monitoring and Management

The health and performance of real servers can be monitored from the FortiGate GUI. Virtual servers and their assigned real servers can be monitored for health status, if there have been any monitor events, number of active sessions, round trip time and number of bytes processed. Should a server become problematic and require administration, it can be gracefully removed from the Real Server pool to enable disruption free maintenance. When a removed real server is able to operate it can gracefully be added back to the virtual server.

HTTP Multiplexing

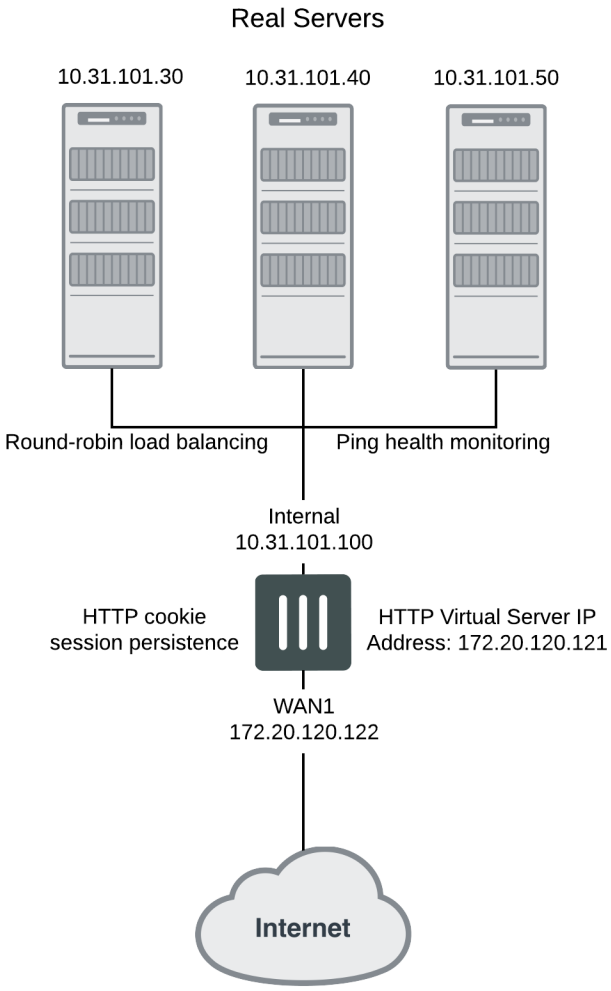
A performance saving feature of HTTP/1.1 compliant web servers is the ability to pipeline requests on the same connection. This allows a single HTTPD process on the server to interleave and server multiple requests. HTTP multiplexing reduces the number idle sessions, too many of which can exhaust the resources on a server. The Fortinet solution has the ability to take multiple separate inbound sessions and multiplex them over the same internal session. This reduces the load on the backend server and increases the overall performance.

Basic load balancing configuration example

This section describes the steps required to configure the load balancing configuration shown below. In this configuration a FortiGate-51B unit is load balancing HTTP traffic from the Internet to three HTTP servers on the Internal network. HTTP sessions are accepted at the wan1 interface with destination IP address 172.20.120.121 on TCP port 8080 and forwarded from the internal interface to the web servers. When forwarded the destination address of the sessions is translated to the IP address of one of the web servers.

The load balancing configuration also includes session persistence using HTTP cookies, round-robin load balancing, and TCP health monitoring for the real servers. Ping health monitoring consists of the FortiGate unit using ICMP ping to make sure the web servers can respond to network traffic.

Virtual server and real servers setup



To configure the example load balancing configuration - general configuration steps

1. Add a load balance ping health check monitor.
   A ping health check monitor causes the FortiGate unit to ping the real servers every 10 seconds. If one of the servers does not respond within 2 seconds, the FortiGate unit will retry the ping 3 times before assuming that the HTTP server is not responding.
2. Add a load balance virtual server.
3. Add the three load balance real servers to the virtual server.
4. Add a security policy that includes the load balance virtual server as the destination address.

To configure the example load balancing configuration

1. Go to **Policy & Objects > Health Check** and add the following health check monitor.

Name	Ping-mon-1
Type	Ping
Interval	10 seconds
Timeout	2 seconds
Retry	3

2. Go to **Policy & Objects > Virtual Servers** and add a virtual server that accepts the traffic to be load balanced.

Name	Vserver-HTTP-1
Type	HTTP
Interface	wan1
Virtual Server IP	172.20.120.121
Virtual Server Port	8080
Load Balance Method	Round Robin
Persistence	HTTP Cookie
Health Check	Ping-mon-1
HTTP Multiplexing	Do not select
Preserve Client IP	Do not select

3. On the same GUI page and the real servers to the virtual server.

IP Address	10.31.101.30
Port	80
Max Connections	0
Mode	Active

IP Address	10.31.101.40
Port	80
Max Connections	0
Mode	Active

IP Address	10.31.101.50
Port	80
Max Connections	0
Mode	Active

4. Go to **Policy & Objects > IPv4 Policy** and add a wan1 to internal security policy that includes the virtual server. This policy also applies an Antivirus profile to the load balanced sessions.

Name	Example-policy
Incoming Interface	wan1
Outgoing Interface	internal
Source	all
Destination	Vserver-HTTP-1
Schedule	always
Service	ALL
Action	ACCEPT
NAT	Turn on NAT and select **Use Outgoing Interface Address**.
Antivirus	Turn on and select an Antivirus profile.

5. Select **OK**.

To configure the example load balancing configuration from the CLI

1. Use the following command to add a Ping health check monitor.

```
config firewall ldb-monitor
   edit ping-mon-1
      set type ping
      set interval 10
      set timeout 2
      set retry 3
   end
```

2. Use the following command to add the virtual server that accepts HTTP sessions on port 8080 at the wan1 interface and load balances the traffic to three real servers.

```
config firewall vip
```

```
   edit Vserver-HTTP-1
      set type server-load-balance
      set server-type http
      set ldb-method round-robin
      set extip 172.20.120.30
      set extintf wan1
      set extport 8080
      set persistence http-cookie
      set monitor tcp-mon-1
         config realservers
            edit 1
               set ip 10.31.101.30
               set port 80
            next
            edit 2
               set ip 10.31.101.40
               set port 80
            end
            edit 3
               set ip 10.31.101.50
               set port 80
            end
end
```

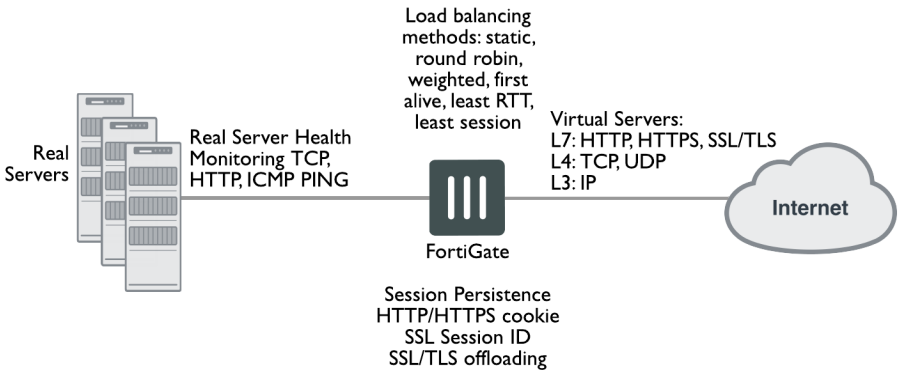
3. Use the following command to add a security policy that includes the load balance virtual server as the destination address.

```
config firewall policy
   edit 0
      set srcintf wan1
      set srcaddr all
      set dstintf internal
      set dstaddr Vserver-HTTP-1
      set action accept
      set schedule always
      set service ALL
      set nat enable
      set utm-status enable
      set profile-protocol-options default
      set av-profile scan
   end
```

Configuring load balancing

This section describes how to use the FortiOS server load balancing to load balance traffic to multiple backend servers.

You can configure FortiOS load balancing to intercept incoming traffic with a virtual server and distribute it among one or more backend real servers. By doing so, FortiOS enables multiple real servers to respond as if they were a single device or virtual server. This in turn means that more simultaneous requests can be handled by the servers.

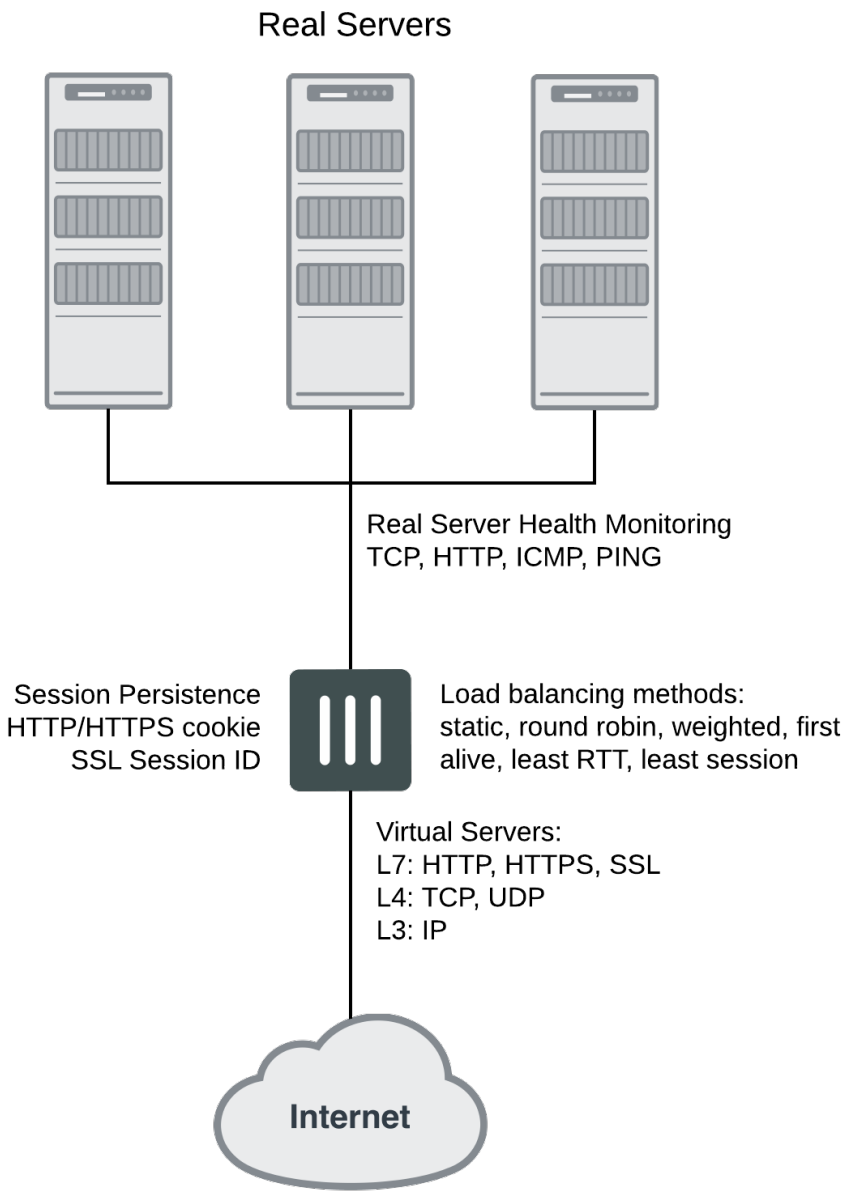


Traffic can be balanced across multiple backend real servers based on a selection of load balancing methods including static (failover), round robin, weighted to account for different sized servers, or based on the health and performance of the server including round trip time, number of connections. The load balancer can balance layer 7 HTTP, HTTPS, SSL, generic layer 4 TCP, UDP and generic layer 3 IP protocols. Session persistence is supported based on injected HTTP/HTTPS cookies or the SSL session ID.

You can bind up to 8 real servers can to one virtual server. The real server topology is transparent to end users, and the users interact with the system as if it were only a single server with the IP address and port number of the virtual server. The real servers may be interconnected by high-speed LAN or by geographically dispersed WAN. The FortiGate unit schedules requests to the real servers and makes parallel services of the virtual server to appear to involve a single IP address.

There are additional benefits to load balancing. First, because the load is distributed across multiple servers, the service being provided can be highly available. If one of the servers breaks down, the load can still be handled by the other servers. Secondly, this increases scalability. If the load increases substantially, more servers can be added behind the FortiGate unit to cope with the increased load.

Server load balancing configuration



Traffic can be balanced across multiple backend real servers based on a selection of load balancing methods including static (failover), round robin, weighted to account for different sized servers, or based on the health and performance of the server including round trip time, number of connections. The load balancer can balance layer 7 HTTP, HTTPS, SSL, generic layer 4 TCP, UDP and generic layer 3 IP protocols. Session persistence is supported based on injected HTTP/HTTPS cookies or the SSL session ID.

You can bind up to 8 real servers can to one virtual server. The real server topology is transparent to end users, and the users interact with the system as if it were only a single server with the IP address and port number of the

virtual server. The real servers may be interconnected by high-speed LAN or by geographically dispersed WAN. The FortiGate unit schedules requests to the real servers and makes parallel services of the virtual server to appear to involve a single IP address.

Load balancing and other FortiOS features

Flow-based and proxy-based security features such as virus scanning, IPS, DLP, application control, and web filtering can be applied to load balanced sessions. This includes SSL offloading and multiplexing. Applying these UTM features to load balancing traffic may reduce load balancing performance.

Authentication is not supported for load balancing sessions. Usually FortiGate load balancing is used to allow public access to services on servers protected by a FortiGate unit. Authentication is not generally not required for this kind of configuration.

Features such web proxying, web caching, and WAN optimization also do not work with load balanced sessions. However, most other features that can be applied by a security policy are supported.

Configuring load balancing from the GUI

A virtual server is a specialized firewall virtual IP that performs server load balancing. From the GUI you add load balancing virtual server by going to **Policy & Objects > Virtual Servers**.

You can use the GUI to configure IPv, IPv6, IPv4 to IPv6 (NAT46), or IPv6 to IPv4 (NAT64) load balancing.

Type

Select the type of virtual server to configure. You can select **IPv4**, **IPv6**, **NAT46**, or **NAT64**. If Type is set to NAT46 or NAT64 you have fewer load balancing options (just HTTP, TCP, UDP and IP) and you can't configure advanced SSL and HTTPS load balancing features.

Name

Enter the name for the virtual server.

Type

Select the protocol to be load balanced by the virtual server. If you select a general protocol such as **IP**, **TCP**, or **UDP** the virtual server load balances all IP, TCP, or UDP sessions. If you select specific protocols such as **HTTP**, **HTTPS**, or **SSL** you can apply additional server load balancing features such as **Persistence** and **HTTP Multiplexing**.

- Select **HTTP** to load balance only HTTP sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced (usually port 80 for HTTP sessions). You can also select **HTTP Multiplex**. You can also set **Persistence** to **HTTP Cookie** to select cookie-based persistence.
- Select **HTTPS** to load balance only HTTPS sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced

(usually port 443 for HTTPS sessions). You can also select **Multiplex HTTP requests/responses.** You can also set **Persistence** to **HTTP Cookie** to select cookie-based persistence. You can also set **Persistence** to **SSL Session ID**.

- Select **IMAPS** to load balance only IMAPS sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced (usually port 993 for IMAPS sessions). You can also set **Persistence** to **SSL Session ID**.
- Select **POP3S** to load balance only POP3S sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced (usually port 995 for POP3S sessions). You can also set **Persistence** to **SSL Session ID**.
- Select **SMTPS** to load balance only SMTPS sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced (usually port 465 for SMTPS sessions). You can also set **Persistence** to **SSL Session ID**.
- Select **SSL** to load balance only SSL sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced.
- Select **TCP** to load balance only TCP sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced.
- Select **UDP** to load balance only UDP sessions with destination port number that matches the **Virtual Server Port** setting. Change **Virtual Server Port** to match the destination port of the sessions to be load balanced.
- Select **IP** to load balance all sessions accepted by the security policy that contains this virtual server.

Interface

Select the virtual server external or outgoing interface from the list. The outgoing interface is connected to the source network and receives the packets to be forwarded to the destination network.

Virtual Server IP

The IPv4 address of the virtual server. This is an IP address on the external interface that you want to map to an address on the destination network.

Virtual Server Port

Enter the external port number that you want to map to a port number on the destination network. Sessions with this destination port are load balanced by this virtual server.

Load Balance Method

Select the load balancing method used by the virtual server.

Persistence

Configure persistence to make sure that a user is connected to the same server every time they make a request that is part of the same session. Session persistence is supported for HTTP and SSL sessions.

Health Check

Select which health check monitor configuration will be used to determine a server's connectivity status.

HTTP Multiplexing

Select to use the FortiGate unit to multiplex multiple client connections into a few connections between the FortiGate unit and the real server.

Preserve Client IP

Select to preserve the IP address of the client in the `X-Forwarded-For` HTTP header. This can be useful if you want log messages on the real servers to the client's original IP address. If this option is not selected, the header will contain the IP address of the FortiGate unit.

This option appears only if **Type** is set to HTTP or HTTPS.

SSL Offloading

Accelerate clients' SSL connections to the server by using the FortiGate to perform SSL operations. This option appears only if **Type** is set to one of the SSL protocols.

Mode

Select which segments of the SSL connection will receive SSL offloading. You can select **Client <-> FortiGate** (or half mode) or **Full** (full mode).

This option appears only if **Type** is set to one of the SSL protocols.

Certificate

Select the certificate to use with **SSL Offloading**. The certificate key size must be 1024 or 2048 bits. 4096-bit keys are not supported.

This option appears only if **Type** is set to one of the SSL protocols.

Real Servers

Add Real Servers to the virtual server. The virtual server load balances traffic to these real servers. See Real servers on page 23.

Configuring load balancing from the CLI

From the CLI you configure IPv4 load balancing by adding a firewall virtual IP and setting the virtual IP type to server load balance:

```
config firewall vip
   edit Vserver-HTTP-1
      set type server-load-balance
      ...
```

Sever load balancing is also supported for:

- IPv6 using the command `config firewall vip6`
- IPv6 to IPv4 using the command `config firewall vip64`
- IPv4 to IPv6 using the commmand `config firewall vip46`

Configuration is the same as IPv4 VIPs except support for advanced HTTP and SSL related features is not available. IPv6 server load balancing supports all the same server types as IPv4 server load balancing (HTTP, HTTPS, IMAPS, POP3S, SMTPS, SSL, TCP, UDP, and IP). IPv4 to IPv6 and IPv6 to IPv4 server load balancing supports fewer server types (HTTP, TCP, UDP, and IP).

A virtual server includes a virtual server IP address bound to an interface. The virtual server IP address is the destination address incoming packets to be load balanced and the virtual server is bound to the interface that receives the packets to be load balanced.

For example, if you want to load balance incoming HTTP traffic from the Internet to a group of web servers on a DMZ network, the virtual server IP address is the known Internet IP address of the web servers and the virtual server binds this IP address to the FortiGate interface connected to the Internet.

When you bind the virtual server's external IP address to a FortiGate unit interface, by default, the network interface responds to ARP requests for the bound IP address. Virtual servers use proxy ARP, as defined in RFC 1027, so that the FortiGate unit can respond to ARP requests on a network for a real server that is actually installed on another network. In some cases you may not want the network interface sending ARP replies. You can use the `arp-reply` option disable sending ARP replies:

```
config firewall vip
   edit Vserver-HTTP-1
      set type server-load-balance
      set arp-reply disable
      ...
```

The load balancing virtual server configuration also includes the virtual server port. This is the TCP port on the bound interface that the virtual server listens for traffic to be load balanced on. The virtual server can listen on any port.

Load balancing methods

The load balancing method defines how sessions are load balanced to real servers. A number of load balancing methods are available as listed below.

All load balancing methods will not send traffic to real servers that are down or not responding. However, the FortiGate unit can only determine if a real server is not responding by using a health check monitor. You should always add at least one health check monitor to a virtual server or to individual real servers, or load balancing methods may attempt to distribute sessions to real servers that are not functioning.

Static

The traffic load is statically spread evenly across all real servers. However, sessions are not assigned according to how busy individual real servers are. This load balancing method provides some persistence because all sessions from the same source address always go to the same real server. However, the distribution is stateless, so if a real server is added or removed (or goes up or down) the distribution is changed and persistence could be lost.

Round Robin

Directs new requests to the next real server, and treats all real servers as equals regardless of response time or number of connections. Dead real servers or non responsive real servers are avoided.

Weighted

Real servers with a higher weight value receive a larger percentage of connections. Set the real server weight when adding a real server.

Least Session

Directs requests to the real server that has the least number of current connections. This method works best in environments where the real servers or other equipment you are load balancing all have similar capabilities. This load balancing method uses the FortiGate session table to track the number of sessions being processed by each real server. The FortiGate unit cannot detect the number of sessions actually being processed by a real server.

Least RTT

Directs sessions to the real server with the least round trip time. The round trip time is determined by a Ping health check monitor and is defaulted to 0 if no Ping health check monitors are added to the virtual server.

First Alive

Always directs sessions to the first alive real server. This load balancing schedule provides real server failover protection by sending all sessions to the first alive real server and if that real server fails, sending all sessions to the next alive real server. Sessions are not distributed to all real servers so all sessions are processed by the “first” real server only.

First refers to the order of the real servers in the virtual server configuration. For example, if you add real servers A, B and C in that order, then all sessions always go to A as long as it is alive. If A goes down then sessions go to B and if B goes down sessions go to C. If A comes back up sessions go back to A. Real servers are ordered in the virtual server configuration in the order in which you add them, with the most recently added real server last. If you want to change the order you must delete and re-add real servers in the required order.

HTTP Host

Load balances HTTP host connections across multiple real servers using the host’s HTTP header to guide the connection to the correct real server.

Session persistence

Use persistence to make sure that a user is connected to the same real server every time they make an HTTP, HTTPS, or SSL request that is part of the same user session. For example, if you are load balancing HTTP and HTTPS sessions to a collection of eCommerce web servers, when a user is making a purchase they will be starting multiple sessions as they navigate the eCommerce site. In most cases all of the sessions started by this user during on eCommerce session should be processed by the same real server. Typically, the HTTP protocol

keeps track of these related sessions using cookies. HTTP cookie persistence makes sure that all sessions that are part of the same user session are processed by the same real server

When you configure persistence, the FortiGate unit load balances a new session to a real server according to the load balance method. If the session has an HTTP cookie or an SSL session ID, the FortiGate unit sends all subsequent sessions with the same HTTP cookie or SSL session ID to the same real server. For more information about HTTP and HTTPS persistence, see "HTTP and HTTPS persistence".

Real servers

Add real servers to a load balancing virtual server to provide the information the virtual server requires to be able to send sessions to the server. A real server configuration includes the IP address of the real server and port number that the real server receives sessions on. The FortiGate unit sends sessions to the real server's IP address using the destination port number in the real server configuration.

When configuring a real server you can also specify the weight (used if the load balance method is set to weighted) and you can limit the maximum number of open connections between the FortiGate unit and the real server. If the maximum number of connections is reached for the real server, the FortiGate unit will automatically switch all further connection requests other real servers until the connection number drops below the specified limit. Setting Maximum Connections to 0 means that the FortiGate unit does not limit the number of connections to the real server.

Real server active, standby, and disabled modes

By default the real server mode setting is active indicating that the real server is available to receive connections. If the real server is removed from the network (for example, for routine maintenance or because of a hardware or software failure) you can change the mode to standby or disabled. In disabled mode the FortiGate unit no longer sends sessions to the real server.

If a real server is in standby mode the FortiGate also does not send sessions to it unless other real servers added to the same virtual server become unavailable. For example:

- A virtual server that includes two real servers one in active mode and one in standby mode. If the real server in active mode fails, the real server in standby mode is changed to active mode and all sessions are sent to this real server.
- A virtual server includes three real servers, two in active mode and one in standby mode, if one of the real servers in active mode fails, the real server in standby mode is changed to active mode and sessions are load balanced between it and still operating real server. If both real servers in active mode fail, all sessions are sent to the real server in standby mode.

Adding real servers from the GUI

To add a real server from the GUI go to **Policy & Objects > Virtual Servers**, edit a virtual server and under **Real Servers** select **Create New** to add a real server to this virtual server.

IP Address

Enter the IP address of the real server.

Port

Enter the port number on the destination network to which the external port number is mapped.

Weight

Enter the weight value of the real server. The higher the weight value, the higher the percentage of connections the server will handle. A range of 1-255 can be used. This option is available only if the associated virtual server's load balance method is **Weighted**.

Max Connections

Enter the limit on the number of active connections directed to a real server. A range of 1-99999 can be used. If the maximum number of connections is reached for the real server, the FortiGate unit will automatically switch all further connection requests to another server until the connection number drops below the specified limit.

Setting **Maximum Connections** to **0** means that the FortiGate unit does not limit the number of connections to the real server.

HTTP Host

Enter the HTTP header for load balancing across multiple real servers. This feature is used for load balancing HTTP host connections across multiple real servers using the host's HTTP header to guide the connection to the correct real server, providing better load balancing for those specific connections.

Mode

Select a mode for the real server. The real server can be active, on standby, or disabled.

Adding real servers from the CLI

To add a real server from the CLI you configure a virtual server and add real servers to it. For example, to add three real servers to a virtual server that load balances UDP sessions on port 8190 using weighted load balancing. For each real server the port is not changed. The default real server port is 0 resulting in the traffic being sent the real server with destination port 8190. Each real sever is given a different weight. Servers with higher weights have a max-connections limit to prevent too many sessions from being sent to them.

```
config firewall vip
   edit Vserver-UDP-1
      set type server-load-balance
      set server-type udp
      set ldb-method weighted
      set extip 172.20.120.30
      set extintf wan1
      set extport 8190
      set monitor ping-mon-1
         config realservers
            edit 1
               set ip 10.31.101.30
               set weight 100
               set max-connections 10000
            next
            edit 2
               set ip 10.31.101.40
               set weight 100
```

```
            set max-connections 10000
        next
        edit 3
            set ip 10.31.101.50
            set weight 10
        end
end
```

Health check monitoring

From the FortiGate GUI you can go to **Policy & Objects > Health Check** and configure health check monitoring so that the FortiGate unit can verify that real servers are able respond to network connection attempts. If a real server responds to connection attempts the load balancer continues to send sessions to it. If a real server stops responding to connection attempts the load balancer assumes that the server is down and does not send sessions to it. The health check monitor configuration determines how the load balancer tests the real servers. You can use a single health check monitor for multiple load balancing configurations.

You can configure TCP, HTTP and Ping health check monitors. Usually you would want the health check monitor to use the same protocol for checking the health of the server as the traffic being load balanced to it. For example, for an HTTP load balancing configuration you would normally use an HTTP health check monitor.

For the TCP and HTTP health check monitors you can specify the destination port to use to connect to the real servers. If you set the port to 0, the health check monitor uses the port defined in the real server. This allows you to use the same health check monitor for multiple real servers using different ports. You can also configure the interval, timeout and retry. A health check occurs every number of seconds indicated by the interval. If a reply is not received within the timeout period the health check is repeated every second. If no response is received after the number of configured retires, the virtual server is considered unresponsive, and load balancing does not srend traffic to that real server. The health check monitor will continue to contact the real server and if successful, the load balancer can resume sending sessions to the recovered real server.

The default health check configuration has an interval of 10 seconds, a timeout of 2 seconds and a retry of 3. This means that the health check monitor checks the health of a real server every 10 seconds. If a reply is not received within 2 seconds the health check monitor re-checks the server every second for 3 retries. If no response is received for 2 seconds after the final retry the server is considered unresponsive. This entire process takes a total of 7 seconds to consider a virtual server as unresponsive (2 second timeout + (3 re-checks x 1 second) + 2 second timeout = 7 seconds). Since this health check process is repeated every 10 seconds, a server can be down for a maximum of 10 + 7 = 17 seconds before the health check monitor considers it down.

For HTTP health check monitors, you can add URL that the FortiGate unit connects to when sending a get request to check the health of a HTTP server. The URL should match an actual URL for the real HTTP servers. The URL is optional.

The URL would not usually include an IP address or domain name. Instead it should start with a “/” and be followed by the address of an actual web page on the real server. For example, if the IP address of the real server is 10.31.101.30, the URL “/test_page.htm” causes the FortiGate unit to send an HTTP get request to “http://10.31.101.30/test_page.htm”.

For HTTP health check monitors, you can also add a matched content phrase that a real HTTP server should include in response to the get request sent by the FortiGate unit using the content of the URL option. If the URL returns a web page, the matched content should exactly match some of the text on the web page. You can use the URL and Matched Content options to verify that an HTTP server is actually operating correctly by responding to get requests with expected web pages. Matched content is only required if you add a URL.

For example, you can set matched content to “server test page” if the real HTTP server page defined by the URL option contains the phrase “server test page”. When the FortiGate unit receives the web page in response to the URL get request, the system searches the content of the web page for the matched content phrase.

Name

Enter the name of the health check monitor configuration.

Type

Select the protocol used to perform the health check.

- TCP
- HTTP
- PING

Port

Enter the port number used to perform the health check. If you set the **Port** to 0, the health check monitor uses the port defined in the real server. This way you can use a single health check monitor for different real servers.

This option does not appear if the **Type** is **PING**.

Interval

Enter the number of seconds between each server health check.

URL

For HTTP health check monitors, add a URL that the FortiGate unit uses when sending a get request to check the health of a HTTP server. The URL should match an actual URL for the real HTTP servers. The URL is optional.

The URL would not usually include an IP address or domain name. Instead it should start with a “/” and be followed by the address of an actual web page on the real server. For example, if the IP address of the real server is 10.10.10.1, the **URL** “/test_page.htm” causes the FortiGate unit to send an HTTP get request to “http://10.10.10.1/test_page.htm”.

This option appears only if **Type** is **HTTP**.

Matched Content

For HTTP health check monitors, add a phrase that a real HTTP server should include in response to the get request sent by the FortiGate unit using the content of the **URL** option. If the **URL** returns a web page, the **Matched Content** should exactly match some of the text on the web page. You can use the **URL** and **Matched Content** options to verify that an HTTP server is actually operating correctly by responding to get requests with expected web pages. Matched content is only required if you add a URL.

For example, you can set **Matched Content** to “server test page” if the real HTTP server page defined by the URL option contains the phrase “server test page”. When the FortiGate unit receives the web page in response to the URL get request, the system searches the content of the web page for the **Matched Content** phrase.

This option appears only if **Type** is **HTTP**.

Max Redirects

For an HTTP health check monitor, specify the maximum number of redirects that the health check monitor will follow when testing the health of the real HTTP server. This feature allows you to do health checking of the HTTP server is accessed through one or more redirects.

Timeout

Enter the number of seconds which must pass after the server health check to indicate a failed health check.

Retry

Enter the number of times, if any, a failed health check will be retried before the server is determined to be inaccessible.

Load balancing limitations

The following limitations apply when adding virtual IPs, load balancing virtual servers, and load balancing real servers. Load balancing virtual servers are actually server load balancing virtual IPs. You can add server load balance virtual IPs from the CLI.

- Virtual IP **External IP Address/Range** entries or ranges cannot overlap with each other or with load balancing virtual server **Virtual Server IP** entries.
- A virtual IP **Mapped IP Address/Range** cannot be 0.0.0.0 or 255.255.255.255.
- A real server **IP** cannot be 0.0.0.0 or 255.255.255.255.
- If a static NAT virtual IP **External IP Address/Range** is 0.0.0.0, the **Mapped IP Address/Range** must be a single IP address.
- If a load balance virtual IP **External IP Address/Range** is 0.0.0.0, the **Mapped IP Address/Range** can be an address range.
- When port forwarding, the count of mapped port numbers and external port numbers must be the same. The GUI does this automatically but the CLI does not.
- Virtual IP and virtual server names must be different from firewall address or address group names.

Monitoring load balancing

From the GUI you can go to **Monitor > Load Balance Monitor** to monitor the status of configured virtual servers and real servers and start or stop the real servers. You can also use the `get test ipldb` command from the CLI to display similar information.

For each real server the monitor displays health status (up or down), active sessions, round trip time (RTT) and the amount of bytes of data processed. From the monitor page you can also stop sending new sessions to any real server. When you select to stop sending sessions the FortiGate unit performs of graceful stop by continuing to send data for sessions that were established or persistent before you selected stop. However, no new sessions are started.

Real Server

The IP addresses of the existing real servers.

Status

Displays the health status according to the health check results for each real server. A green arrow means the server is up. A red arrow means the server is down.

Mode

The mode of the health check monitor. Can be active, standby, or disabled.

Monitor Events

Display each real server's up and down times.

Active Sessions

Display each real server's active sessions.

RTT (ms)

Displays the Round Trip Time (RTT) of each real server. By default, the RTT is "<1". This value will change only when ping monitoring is enabled on a real server.

Bytes Processed

Displays the traffic processed by each real server.

Graceful Stop/Start

Select to start or stop real servers. When stopping a server, the FortiGate unit will not accept new sessions but will wait for the active sessions to finish.

Load balancing diagnose commands

You can also use the following diagnose commands to view status information for load balancing virtual servers and real servers:

```
diagnose firewall vip realserver {down | healthcheck | list | up}
diagnose firewall vip virtual-server {filter | real-server | stats}
```

For example, the following command lists and displays status information for all real servers:

```
diagnose firewall vip virtual-server real-server

vd root/0 vs vs/2 addr 10.31.101.30:80 status 1/1
conn: max 0 active 0 attempts 0 success 0 drop 0 fail 0
```

```
vd root/0 vs vs/2 addr 10.31.101.20:80 status 1/1
conn: max 0 active 0 attempts 0 success 0 drop 0 fail 0
```

Many of the diagnostic commands involve retrieving information about one or more virtual servers. To control which servers are queried you can define a filter:

```
diagnose firewall vip virtual-server filter <filter_str>
```

Where `<filter_str>` can be:

- `clear` erase the current filter
- `dst` the destination address range to filter by
- `dst-port` the destination port range to filter by
- `list` display the current filter
- `name` the vip name to filter by
- `negate` negate the specified filter parameter
- `src` the source address range to filter by
- `src-port` the source port range to filter by
- `vd` index of virtual domain. -1 matches all

The default filter is empty so no filtering is done.

Logging diagnostics

The logging diagnostics provide information about two separate features:

```
diagnose firewall vip virtual-server filter
```

- `filter` sets a filter for the virtual server debug log
- The filter option controls what entries the virtual server daemon will log to the console if `diagnose debug application vs` level is non-zero. The filtering can be done on source, destination, virtual-server name, virtual domain, and so on:

```
diagnose firewall vip virtual-server filter <filter_str>
```

Where `<filter_str>` can be

- `clear` erase the current filter
- `dst` the destination address range to filter by
- `dst-port` the destination port range to filter by
- `list` display the current filter
- `name` the virtual-server name to filter by
- `negate` negate the specified filter parameter
- `src` the source address range to filter by
- `src-port` the source port range to filter by
- `vd` index of virtual domain. -1 matches all

The default filter is empty so no filtering is done.

Real server diagnostics

Enter the following command to list all the real servers:

```
diagnose firewall vip virtual-server real-server list
```

In the following example there is only one virtual server called `slb` and it has two real-servers:

```
diagnose firewall vip virtual-server server
```

```
vd root/0 vs slb/2 addr 172.16.67.191:80 status 1/1
conn: max 10 active 0 attempts 0 success 0 drop 0 fail 0
http: available 0 total 0

vd root/0 vs slb/2 addr 172.16.67.192:80 status 1/1
conn: max 10 active 1 attempts 4 success 4 drop 0 fail 0
http: available 1 total 1
```

The `status` indicates the administrative and operational status of the real-server.

- `max` indicates that the real-server will only allow 10 concurrent connections.
- `active` is the number of current connections to the server attempts is the total number of connections attempted success is the total number of connections that were successful.
- `drop` is the total number of connections that were dropped because the active count hit max.
- `fail` is the total number of connections that failed to complete due to some internal problem (for example, lack of memory).

If the virtual server has HTTP multiplexing enabled then the HTTP section indicates how many established connections to the real-sever are available to service a HTTP request and also the total number of connections.

HTTP and HTTPS load balancing, multiplexing, and persistence

In a firewall load balancing virtual server configuration, you can select HTTP to load balance only HTTP sessions. The virtual server will load balance HTTP sessions received at the virtual server interface with destination IP address that matches the configured virtual server IP and destination port number that matches the configured virtual server port. The default virtual server port for HTTP load balancing is 80, but you can change this to any port number. Similarly for HTTPS load balancing, set the virtual server type to HTTPS and then select the interface, virtual server IP, and virtual server port that matches the HTTPS traffic to be load balanced. Usually HTTPS traffic uses port 443.

You can also configure load balancing to offload SSL processing for HTTPS and SSL traffic. See SSL/TLS load balancing on page 36.

HTTP and HTTPS multiplexing

For both HTTP and HTTPS load balancing you can multiplex HTTP requests and responses over a single TCP connection. HTTP multiplexing is a performance saving feature of HTTP/1.1 compliant web servers that provides the ability to pipeline many unrelated HTTP or HTTPS requests on the same connection. This allows a single HTTPD process on the server to interleave and serve multiple requests. The result is fewer idle sessions on the web server so server resources are used more efficiently. HTTP multiplexing can take multiple separate inbound sessions and multiplex them over the same internal session. This may reduce the load on the backend server and increase the overall performance.

HTTP multiplexing may improve performance in some cases. For example, if users web browsers are only compatible with HTTP 1.0. HTTP multiplexing can also improve performance between a web server and the FortiGate unit if the FortiGate unit is performing SSL acceleration. However, in most cases HTTP multiplexing should only be used if enabling it leads to a measurable improvement in performance.

To enable HTTP multiplexing from the GUI, select multiplex HTTP requests/responses over a single TCP connection. To enable HTTP multiplexing from the CLI enable the `http-multiplex` option.

Preserving the client IP address

Select preserve client IP from the GUI or enable the `http-ip-header` option from the CLI to preserve the IP address of the client in the `X-Forwarded-For` HTTP header. This can be useful in an HTTP multiplexing configuration if you want to be able to see the original client IP address in log messages on the destination web server. If this option is not selected, the `X-Forwarded-For` HTTP header contains the IP address of the FortiGate unit.

Preserving the client IP address in a different HTTP header

If you select preserve client IP from the GUI or enable the `http-ip-header` option from the CLI you can also preserve the client IP in a different HTTP header. This can be useful if you want to use a custom header name instead of `X-Forwarded-For`.

You can add the custom header name from the CLI. When `http-ip-header` is enabled you can add a custom header name to the `http-ip-header-name` option. If you don't add a name the `X-Forwarded-For` header is used.

HTTP and HTTPS persistence

Configure load balancing persistence for HTTP or HTTPS to make sure that a user is connected to the same server every time they make a request that is part of the same session. HTTP cookie persistence uses injected cookies to enable persistence.

When you configure persistence, the FortiGate unit load balances a new session to a real server according to the **Load Balance Method**. If the session has an HTTP cookie or an SSL session ID, the FortiGate unit sends all subsequent sessions with the same HTTP cookie or SSL session ID to the same real server.

The following example shows how to enable cookie persistence and set the cookie domain to .example.org.

```
config firewall vip
   edit HTTP_Load_Balance
      set type server-load-balance
      set server-type http
      set extport 8080
      set extintf port2
      set extip 192.168.20.20
      set persistence http-cookie
      set http-cookie-domain .example.org
         config realservers
         edit 1
               set ip 10.10.10.1
               set port 80
         next
         edit 2
               set ip 10.10.10.2
               set port 80
         next
         edit 3
               set ip 10.10.10.3
               set port 80
         end
```

How HTTP cookie persistence options work

The following options are available for the `config firewall vip` command when `type` is set to `server-load-balance`, `server-type` is set to `http` or `https` and `persistence` is set to `http-cookie`:

```
http-cookie-domain-from-host
http-cookie-domain
http-cookie-path
http-cookie-generation
http-cookie-age
http-cookie-share
https-cookie-share
```

When HTTP cookie persistence is enabled the FortiGate unit inserts a header of the following form into each HTTP response unless the corresponding HTTP request already contains a `FGTServer` cookie:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1; Max-Age=3600
```

The value of the `FGTServer` cookie encodes the server that traffic should be directed to. The value is encoded so as to not leak information about the internal network.

Enable `http-cookie-domain-from-host` to extract the cookie domain from the `host:` header in the HTTP request. For example, to restrict the cookie to `.server.com`, enter:

The generated cookies could have the following form if the **Host:** header contains **exhost.com**:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1;
    Domain=.exhost.com; Max-Age=3600
```

For more information, see "HTTP host-based load balancing".

Use `http-cookie-domain` to restrict the domain that the cookie should apply to. For example, to restrict the cookie to `.server.com`, enter:

```
set http-cookie-domain .server.com
```

Now all generated cookies will have the following form:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1;
    Domain=.server.com; Max-Age=3600
```

Use `http-cookie-path` to limit the cookies to a particular path. For example, to limit cookies to the path `/sales`, enter:

```
set http-cookie-path /sales
```

Now all generated cookies will have the following form:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1;
    Domain=.server.com; Path=/sales; Max-Age=3600
```

Use `http-cookie-age` to change how long the browser caches the cookie. You can enter an age in minutes or set the age to 0 to make the browser keep the cookie indefinitely:

```
set http-cookie-age 0
```

Now all generated cookies will have the following form:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1;
    Domain=.server.com; Path=/sales
```

Use `http-cookie-generation` to invalidate all cookies that have already been generated. The exact value of the generation is not important, only that it is different from any generation that has already been used for cookies in this domain. The simplest approach is to increment the generation by one each time invalidation is required. Since the default is 0, enter the following to invalidate all existing cookies:

```
set http-cookie-generation 1
```

Use `http-cookie-share {disable | same-ip}` to control the sharing of cookies across virtual servers in the same virtual domain. The default setting `same-ip` means that any `FGTServer` cookie generated by one virtual server can be used by another virtual server in the same virtual domain. For example, if you have an application that starts on HTTP and then changes to HTTPS and you want to make sure that the same server is used for the HTTP and HTTPS traffic then you can create two virtual servers, one for port 80 (for HTTP) and one for port 443 (for HTTPS). As long as you add the same real servers to both of these virtual servers (and as long as both virtual servers have the same number of real servers with the same IP addresses), then cookies generated by accessing the HTTP server are reused when the application changes to the HTTPS server.

If for any reason you do not want this sharing to occur then select `disable` to make sure that a cookie generated for a virtual server cannot be used by other virtual servers.

Use `https-cookie-secure` to enable or disable using secure cookies. Secure cookies are disabled by default because secure cookies can interfere with cookie sharing across HTTP and HTTPS virtual servers. If enabled, then the `Secure` tag is added to the cookie inserted by the FortiGate unit:

```
Set-Cookie: FGTServer=E7D01637C4B08E89A6714213A9D85D9C7E4D8158; Version=1; Max-Age=3600;
    Secure
```

HTTP host-based load balancing

When configuring HTTP or HTTPS load balancing you can select HTTP host load balancing to load balance HTTP host connections across multiple real servers using the host's HTTP header to guide the connection to the correct real server. HTTP 1.1 includes the concept of a virtual server which allows a HTTP or HTTPS server with a single external IP address to serve requests for multiple DNS domains by using the mandatory `Host:` header in a HTTP request to indicate which DNS domain the request is destined for.

FortiOS can load-balance HTTP and HTTPS connections among multiple real servers using the `Host:` header to guide the connection to the correct real server. The host load balancing method allows a real server to specify a `http-host` attribute which is the domain name of the traffic for that real server. Each real server can only specify a single domain name. The same domain name can appear in more than one real server but only the first one that is up will be used, any others are purely for redundancy. If the Host: header contains a domain that does not match any http-host entry then the connection will be dropped. A real server with no http-host can be matched by any Host: domain.

For example, consider a FortiGate unit that is load-balancing traffic to three real servers. Traffic for www.example1.com should go to 192.168.2.1, traffic for www.example2.com should go to 192.168.2.2 and traffic to any other domain should go to 192.168.2.3. To enable this configuration you would add a virtual server and set the load balance method to HTTP host. Then you would add three real servers and set the HTTP host of the real server with IP address 192.168.2.1 to www.example1.com, the HTTP host of the real server with IP address 192.168.2.2 to www.example2.com and you would not specify an HTTP host for the third real server.

The configuration of a virtual IP to achieve this result would be:

```
config firewall vip
   edit "http-host-ldb"
      set type server-load-balance
      set extip 172.16.67.195
      set extintf "lan"
      set server-type http
      set ldb-method http-host
      set extport 80
         config realservers
         edit 1
               set http-host "www.example1.com"
               set ip 192.168.2.1
               set port 80
         next
         edit 2
               set http-host "www.example2.com"
               set ip 192.168.2.2
               set port 80
         next
         edit 3
               set ip 192.168.2.3
               set port 80
         next
         end
end
```

Host load balancing and HTTP cookie persistence

In an HTTP host-based load balancing configuration with HTTP cookie persistence enabled you can optionally configure cookie persistence to use the domain set in the host header as the cookie domain. You can do this by enabling the `http-cookie-domain-from-host` option, for example:

```
config firewall vip
   edit "http-host-ldb"
      set type server-load-balance
      set extip 172.16.67.195
      set extintf "lan"
      set server-type http
      set ldb-method http-host
      set extport 80
      set persistence http-cookie
      set http-cookie-domain-from-host enable
         config realservers
            edit 1
               set http-host "www.example1.com"
               set ip 192.168.2.1
               set port 80
            next
            edit 2
               set http-host "www.example2.com"
               set ip 192.168.2.2
               set port 80
            next
            edit 3
               set ip 192.168.2.3
               set port 80
            next
            end
end
```

SSL/TLS load balancing

In a firewall load balancing virtual server configuration, you can select SSL to load balance only SSL and TLS sessions. The virtual server will load balance SSL and TLS sessions received at the virtual server interface with destination IP address that matches the configured virtual server IP and destination port number that matches the configured virtual server port. Change this port to match the destination port of the sessions to be load balanced.

For SSL load balancing you can also set persistence to SSL session ID. Persistence is achieved by the FortiGate unit sending all sessions with the same SSL session ID to the same real server. When you configure persistence, the FortiGate unit load balances a new session to a real server according to the **Load Balance Method**. If the session has an SSL session ID, the FortiGate unit sends all subsequent sessions with the same SSL session ID to the same real server.

SSL/TLS offloading

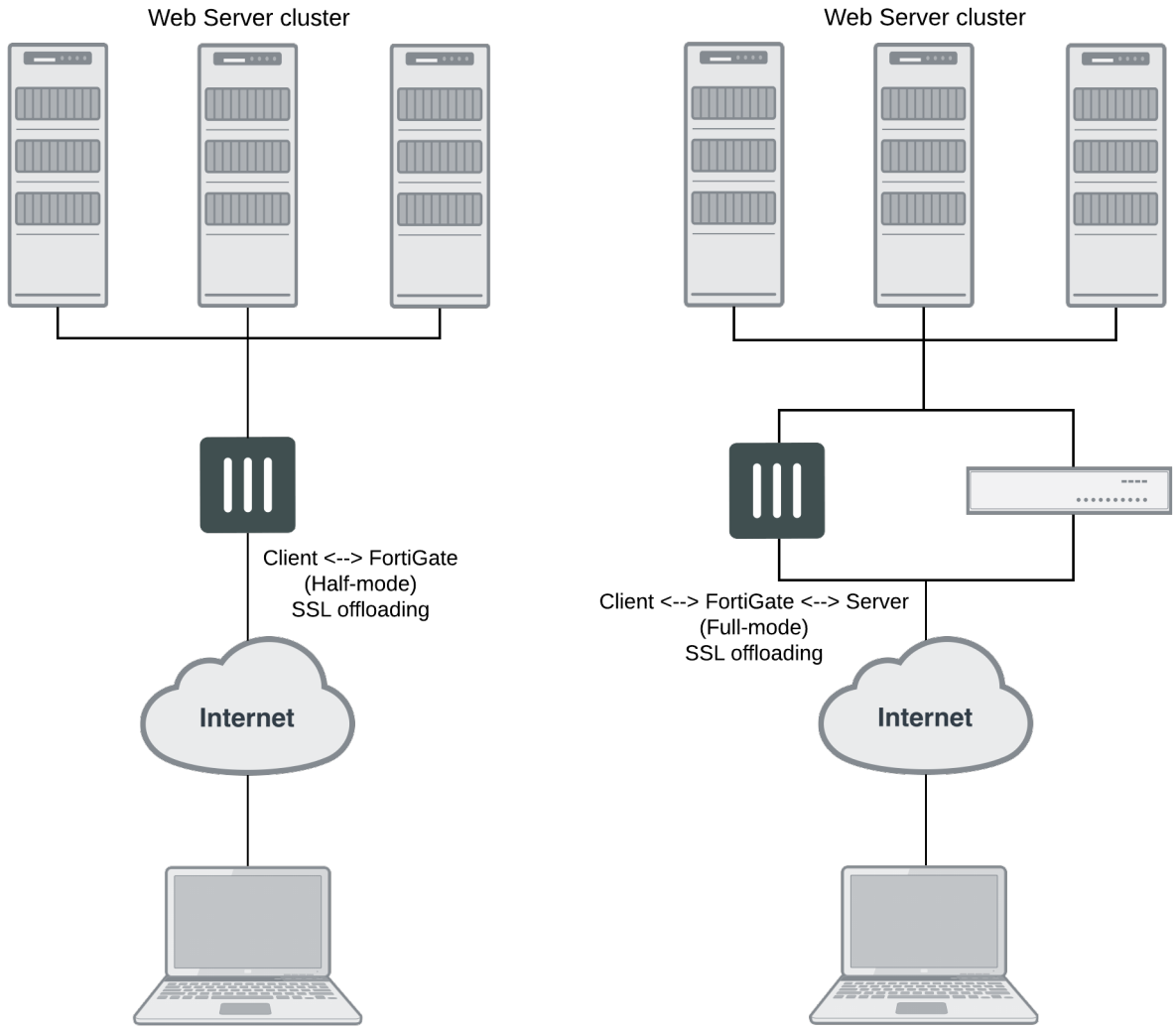
Use SSL offloading to accelerate clients' SSL or HTTPS connections to real servers by using the FortiGate unit to perform SSL/TLS operations (offloading them from the real servers using the FortiGate unit's SSL acceleration hardware). FortiGate units can offload most versions of SSL/TLS, including SSL 3.0, TLS 1.0 and TLS 1.2. SSL/TLS offloading is available on FortiGate units that support SSL acceleration.

To configure SSL offloading from the GUI go to **Policy & Objects > Virtual Servers**. Add a virtual server and set the type to HTTPS or SSL and select the SSL offloading type (**Client <-> FortiGate** or **Full**).

Select **Client <-> FortiGate** to apply hardware accelerated SSL/TLS processing only to the part of the connection between the client and the FortiGate unit. This mode is called half mode SSL offloading. The segment between the FortiGate unit and the server will use clear text communications. This results in best performance, but cannot be used in failover configurations where the failover path does not have an SSL accelerator.

Select **Full** to apply hardware accelerated SSL processing to both parts of the connection: the segment between client and the FortiGate unit, and the segment between the FortiGate unit and the server. The segment between the FortiGate unit and the server uses encrypted communications, but the handshakes are abbreviated. This is not as efficient as half mode SSL offloading, but still improves performance. As well, full-mode SSL offloading can be used in failover configurations where the failover path does not have an SSL accelerator. If the server is already configured to use SSL, this also enables SSL acceleration without requiring changes to the server's configuration.

SSL Offloading modes (Half Mode and Full Mode)



Configuring SSL offloading also requires selecting a certificate to use for the SSL offloading sessions. SSL offloading supports key sizes up to 4096. FortiGate models with CP9 processors support 3072 and 4096 DH bit sizes in hardware. All FortiGate models up to and including those with CP8 processors only support offloading DH bit sizes up to 2048 so any sizes larger than that are done in software and thus are relatively resource intensive

The following CLI command shows an example half mode HTTPS SSL offloading configuration. In the example the `ssl-mode` option sets the SSL offload mode to `half` (which is the default mode).

```
config firewall vip
   edit Vserver-ssl-offload
      set type server-load-balance
      set server-type https
      set ldb-method round-robin
      set extip 172.20.120.30
      set extintf wan1
      set extport 443
```

```
        set persistence ssl-session-id
        set ssl-mode half
        set ssl-certificate my-cert
        set monitor tcp-mon-1
           config realservers
              edit 1
                 set ip 10.31.101.30
                 set port 443
              next
              edit 2
                 set ip 10.31.101.40
                 set port 443
              end
    end
```

Separate virtual-server client and server TLS version and cipher configuration

In some cases, you may want the to use different versions of SSL or TLS on the client to FortiGate connection than on the FortiGate to server connection. For example, you may want to use the FortiGate to protect a legacy SSL 3.0 or TLS 1.0 server while making sure that client to FortiGate connections must always use the higher level of protection offered by TLS 1.1 or greater. Also, in some cases you might want to protect a server that only has weak ciphers (for example, DES or RC4) while making sure that all connections between the FortiGate and the client use a strong cipher for better protection.

The following options are available when configuring server load balancing for HTTPS sessions configured with the following command:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-mode full
      ...
```

Setting the SSL/TLS versions to use for server and client connections

The `ssl-server-min-version, ssl-server-max-version`, `ssl-min-version` and `ssl-max-version` configuration options allow the minimum and maximum SSL/TLS versions for the client to FortiGate connection to be independent of the FortiGate to server configuration. By default these options are both set to `client` and the configured `ssl-min-version` and `ssl-max-version` settings are applied to both the client and the server connection.

You can change the `ssl-server-min-version` and `ssl-server-max-version` to apply different options to the server connection. The `ssl-min-version` and `ssl-max-version` settings are still applied to the client connection. If you set the `ssl-server-min-version` and `ssl-server-max-version` to an explicit version then both must be set to an explicit version.

The `ssl-server-min-version` and `ssl-server-max-version` options allow you to specify the minimum and maximum SSL/TLS versions the FortiGate will offer to the server (in the record header of the ClientHello) when performing full mode SSL offloading and thus the minimum and maximum SSL/TLS versions the FortiGate accepts from the server (in a ServerHello). If the server responds with a version in its ServerHello

that is lower than `ssl-server-min-version` or higher than the `ssl-server-max-version` then the FortiGate terminates the connection.

Command syntax is:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-mode full
      set ssl-min-version {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2}
      set ssl-max-version {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2}
      set ssl-server-min-version {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2 | client}
      set ssl-server-max-version {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2 | client}
```

Setting the SSL/TLS cipher choices for server and client connections

The `ssl-algorithm` and `ssl-server-algorithm` configuration options allow the cipher choice for the FortiGate to server connection to be independent of the client to FortiGate connection. By default, `ssl-server-algorithm` is set to `client` and the configured `ssl-algorithm` setting is applied to both the client and the server connection.

You can change the `ssl-server-algorithm` to apply different options to the server connection. The `ssl-algorithm` setting is still applied to the client connection.

The following `ssl-server-algorithm` options are available:

- `high`, offer AES or 3DES cypher suites in the ServerHello
- `medium`, use AES, 3DES, or RC4 cypher suites in the ServerHello
- `low`, use AES, 3DES, RC4, or DES cypher suites in the ServerHello
- `custom`, specifiy custom cypher suites using the `config ssl-server-cipher-suites` and offer these custom cypher suites in the ServerHello.
- `client`, offer the cypher suites in the ServerHello that are offered in the ClientHello.

Command syntax is:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-mode full
      set ssl-algorithm {high | medium | low | custom}
      set ssl-server-algorithm {high | medium | low | custom | client}
```

If you set ssl-server-algorithm to custom, the syntax is:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-mode full
      set ssl-server-algorithm custom
         config ssl-server-cipher-suites
            edit 10
               set cipher <cipher-suite>
               set versions {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2}
```

```
next
edit 20
   set cipher <cipher-suite>
   set versions {ssl-3.0 | tls-1.0 | tls-1.1 | tls-1.2}
end
```

Protection from TLS protocol downgrade attacks

The `ssl-client-fallback` option, when enabled (the default configuration), performs downgrade attack prevention (RFC 7507).

Command syntax is:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-client-fallback {disable | enable}
```

Setting 3072- and 4096-bit Diffie-Hellman values

The `ssl-dh-bits` option allows you to specify the number of bits of the prime number used in the Diffie-Hellman exchange for RSA encryption of the SSL connection. Larger prime numbers are associated with greater cryptographic strength. You can set DH values from 768 to 4096 bits.

Command syntax is:

```
config firewall vip
   edit server-name
      set type server-load-balance
      set server-type https
      set ssl-dh-bits {768 | 1024 | 1536 | 2048 | 3072 | 4096}
```

Setting the DH bits to 2048 only provides the equivalent of a symmetric cipher in the range of 112 - 128 bits. This means that if AES 256 is used then the weakest point is the DH of 2048 and a value of at least 3072 should be use if the goal is to have 256 bits of security.

FortiGate models with CP9 processors support 3072 and 4096 DH bit sizes in hardware. All FortiGate models up to and including those with CP8 processors only support offloading DH bit sizes up to 2048 so any sizes larger than that are done in software and thus are relatively resource intensive.

Additional SSL load balancing and SSL offloading options

The following SSL load balancing and SSL offloading options are only available from the CLI:

```
ssl-client-session-state-max <sessionstates_int>
```

Enter the maximum number of SSL session states to keep for the segment of the SSL connection between the client and the FortiGate unit.

```
ssl-client-session-state-timeout <timeout_int>
```

Enter the number of minutes to keep the SSL session states for the segment of the SSL connection between the client and the FortiGate unit.

```
ssl-client-session-state-type {both | client | disable | time}
```

Select which method the FortiGate unit should use when deciding to expire SSL sessions for the segment of the SSL connection between the client and the FortiGate unit.

- `both`: Select to expire SSL session states when either `ssl-client-session-state-max` or `ssl-client-session-state-timeout` is exceeded, regardless of which occurs first.
- `count`: Select to expire SSL session states when `ssl-client-session-state-max` is exceeded.
- `disable`: Select to keep no SSL session states.
- `time`: Select to expire SSL session states when `ssl-client-session-state-timeout` is exceeded.

```
ssl-http-location-conversion {enable | disable}
```

Select to replace `http` with `https` in the reply's `Location` HTTP header field. For example, in the reply, `Location: http://example.com/` would be converted to `Location: https://example.com/`

```
ssl-http-match-host {enable | disable}
```

Enable (the default) to apply `Location` conversion to the reply's HTTP header only if the host name portion of `Location` matches the request's `Host` field, or, if the `Host` field does not exist, the host name portion of the request's URI.

If disabled, conversion occurs regardless of whether the host names in the request and the reply match.

For example, if host matching is enabled, and a request contains `Host: example.com` and the reply contains `Location: http://example.cc/`, the `Location` field does not match the host of the original request and the reply's `Location` field remains unchanged. If the reply contains `Location: http://example.com/`, however, then the FortiGate unit detects the matching host name and converts the reply field to `Location: https://example.com/`.

This option appears only if `ssl-http-location-conversion` is `enable`.

```
ssl-send-empty-frags {enable | disable}
```

Select to precede the record with empty fragments to protect from attacks on CBC IV. You might disable this option if SSL acceleration will be used with an old or buggy SSL implementation which cannot properly handle empty fragments.

```
ssl-server-session-state-max <sessionstates_int>
```

Enter the maximum number of SSL session states to keep for the segment of the SSL connection between the server and the FortiGate unit.

```
ssl-server-session-state-timeout <timeout_int>
```

Enter the number of minutes to keep the SSL session states for the segment of the SSL connection between the server and the FortiGate unit. This option appears only if `ssl-mode` is `full`.

```
ssl-server-session-state-type {both | count | disable | time}
```

Select which method the FortiGate unit should use when deciding to expire SSL sessions for the segment of the SSL connection between the server and the FortiGate unit. This option appears only if `ssl-mode` is `full`.

- `both`: Select to expire SSL session states when either `ssl-server-session-state-max` or `ssl-server-session-state-timeout` is exceeded, regardless of which occurs first.
- `count`: Select to expire SSL session states when `ssl-server-session-state-max` is exceeded.
- `disable`: Select to keep no SSL session states.
- `time`: Select to expire SSL session states when `ssl-server-session-state-timeout` is exceeded.

SSL offloading support for Internet Explorer 6

In some cases the Internet Explorer 6 web browser may be able to access real servers. To resolve this issue, disable the `ssl-send-empty-frags` option:

```
config firewall vip
   edit vip_name
      set type server-load-balance
      set server-type https
      set ssl-send-empty-frags disable
   end
```

You can disable this option if SSL acceleration will be used with an old or buggy SSL implementation that cannot properly handle empty fragments.

Selecting the cipher suites available for SSL load balancing

You can use the following command to view the complete list of cipher suites available for SSL offloading:

```
config firewall vip
   edit <vip-name>
      set type server-load-balance
      set server-type https
      set ssl-algorithm custom
         config ssl-cipher-suites
            edit 0
            set cipher ?
```

In most configurations the matching cipher suite is automatically selected but you can limit the set of cipher suites that are available for a given SSL offloading configuration. For example, use the following command to limit an SSL load balancing configuration to use the three cipher suites that support ChaCha20 and Poly1305:

```
config firewall vip
   edit <vip-name>
      set type server-load-balance
      set server-type https
      set ssl-algorithm custom
         config ssl-cipher-suites
            edit 1
            set cipher TLS-ECDHE-RSA-WITH-CHACHA20-POLY1305-SHA256
            next
            edit 2
            set cipher TLS-ECDHE-ECDSA-WITH-CHACHA20-POLY1305-SHA256
            next
            edit 3
            set cipher TLS-DHE-RSA-WITH-CHACHA20-POLY1305-SHA256
            end
         end
```

Disabling SSL/TLS re-negotiation

The vulnerability CVE-2009-3555 affects all SSL/TLS servers that support re-negotiation. FortiOS when configured for SSL/TLS offloading is operating as a SSL/TLS server. The IETF is working on a TLS protocol change that will fix the problem identified by CVE-2009-3555 while still supporting re-negotiation. Until that protocol change is available, you can use the `ssl-client-renegotiation` option to disable support for SSL/TLS re-negotiation. The default value of this option is `allow`, which allows an SSL client to renegotiate. You can change the setting to `deny` to abort any attempts by an SSL client to renegotiate. If you select `deny` as soon as a `ClientHello` message indicating a re-negotiation is received from the client FortiOS terminates the TCP connection.

Since SSL offloading does not support requesting client certificates the only circumstance in which a re-negotiation is required is when more than 2^32 bytes of data are exchanged over a single handshake. If you are sure that this volume of traffic will not occur then you can disable re-negotiation and avoid any possibility of the attack described in CVE-2009-3555.

The re-negotiation behavior can be tested using OpenSSL. The OpenSSL s_client application has the feature that the user can request that it do renegotiation by typing “R”. For example, the following shows a successful re-negotiation against a FortiGate unit configured with a VIP for 192.168.2.100:443:

```
$ openssl s_client -connect 192.168.2.100:443
   CONNECTED(00000003)
   depth=1 /C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
   Authority/CN=support/emailAddress=support@fortinet.com
   verify error:num=19:self signed certificate in certificate chain
   verify return:0
   ---
   Certificate chain
   0

     s:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Fortigate/CN=FW80CM3909604325/emailAdd
     ress=support@fortinet.com
       i:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
       Authority/CN=support/emailAddress=support@fortinet.com
   1 s:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
   Authority/CN=support/emailAddress=support@fortinet.com
      i:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
      Authority/CN=support/emailAddress=support@fortinet.com
   ---
   Server certificate
-----BEGIN CERTIFICATE-----

---certificate not shown---

-----END CERTIFICATE-----

     subject=/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Fortigate/CN=FW80CM3909604325/em
     ailAddress=support@fortinet.com
   issuer=/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
   Authority/CN=support/emailAddress=support@fortinet.com
   ---
   No client certificate CA names sent
   ---
   SSL handshake has read 2370 bytes and written 316 bytes
   ---
```

```
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 1024 bit
Compression: NONE
Expansion: NONE
SSL-Session:
   Protocol : TLSv1
   Cipher : DHE-RSA-AES256-SHA
   Session-ID:
   02781E1E368DCCE97A95396FAA82E8F740F5BBA96CF022F6FEC3597B0CC88095
   Session-ID-ctx:
   Master-Key:

 A6BBBD8477A2422D56E57C1792A4EA9C86F37D731E67D0A66E5CDB2B5C76650780C0E7F01CFF851EC44661
 86F4C48397
   Key-Arg : None
   Start Time: 1264453027
   Timeout : 300 (sec)
   Verify return code: 19 (self signed certificate in certificate
   chain)
---
GET /main.c HTTP/1.0
R
RENEGOTIATING
depth=1 /C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
Authority/CN=support/emailAddress=support@fortinet.com
verify error:num=19:self signed certificate in certificate chain
verify return:0
HTTP/1.0 200 ok
Content-type: text/plain

/*
* Copyright (C) 2004-2007 Fortinet
*/

#include <stdio.h>
#include "vsd_ui.h"

int main(int argc, char **argv)
{
return vsd_ui_main(argc, argv);
}
closed
$
```

The following is the same test, but this time with the VIP configuration changed to `ssl-client-renegotation deny`:

```
$ openssl s_client -connect 192.168.2.100:443
CONNECTED(00000003)
depth=1 /C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
Authority/CN=support/emailAddress=support@fortinet.com
verify error:num=19:self signed certificate in certificate chain
verify return:0
---
Certificate chain
0
```

```
     s:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Fortigate/CN=FW80CM3909604325/emailAdd
     ress=support@fortinet.com
       i:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
       Authority/CN=support/emailAddress=support@fortinet.com
   1 s:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
   Authority/CN=support/emailAddress=support@fortinet.com
      i:/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
      Authority/CN=support/emailAddress=support@fortinet.com
   ---
   Server certificate
   -----BEGIN CERTIFICATE-----
---certificate not shown---
-----END CERTIFICATE-----

    subject=/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Fortigate/CN=FW80CM3909604325/em
    ailAddress=support@fortinet.com
   issuer=/C=US/ST=California/L=Sunnyvale/O=Fortinet/OU=Certificate
   Authority/CN=support/emailAddress=support@fortinet.com
   ---
   No client certificate CA names sent
   ---
   SSL handshake has read 2370 bytes and written 316 bytes
   ---
   New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
   Server public key is 1024 bit
   Compression: NONE
   Expansion: NONE
   SSL-Session:
      Protocol : TLSv1
      Cipher : DHE-RSA-AES256-SHA
      Session-ID:
      8253331D266DDE38E4D8A04AFCA9CBDED5B1134932CE1718EED6469C1FBC7474
      Session-ID-ctx:
      Master-Key:

    ED05A3EF168AF2D06A486362FE91F1D6CAA55CEFC38A3C36FB8BD74236BF2657D4701B6C1456CEB5BB5EFA
    A7619EF12D
      Key-Arg : None
      Start Time: 1264452957
      Timeout : 300 (sec)
      Verify return code: 19 (self signed certificate in certificate
      chain)
   ---
   GET /main.c HTTP/1.0
   R
   RENEGOTIATING
19916:error:1409E0E5:SSL routines:SSL3_WRITE_BYTES:ssl handshake failure:s3_pkt.c:530:
```

Use the following command to check the SSL stats to see that the `renegotiations blocked` counter is now 1:

```
diagnose firewall vip virtual-server stats ssl
   ssl
      client
         connections total 0 active 0 max 0
         handshakes total 4 active 0 max 0 completed 4 abbreviated 0
         session states total 4 active 4 max 4
         cipher-suite failures 0
         embryonics total 0 active 0 max 0 terminated 0
```

```
            renegotiations blocked 1
         server
            connections total 0 active 0 max 0
            handshakes total 3 active 0 max 0 completed 2 abbreviated 1
            session states total 1 active 1 max 1
            cipher-suite failures 0
         internal error 0
         bad handshake length 0
         bad change cipher spec length 0
         pubkey too big 0
         persistence
            find 0 found 0 clash 0 addr 0 error 0
```

If the virtual server debug log is examined (diagnose debug appl vs -1) then at the point the re-negotiation is blocked there is a log:

```
vs ssl 12 handshake recv ClientHello
   vs ssl 12 handshake recv 1
       (0100005403014b5e056c7f573a563bebe0258c3254bbaff7046a461164f34f94f4f3d019c418000026
       003900380035001600130 00a00330032002f0005000400150012000900140011000800060003020100
       00400230000)
   vs ssl 12 client renegotiation attempted rejected, abort
   vs ssl 12 closing 0 up
   vs src 12 close 0 in
   vs src 12 error closing
   vs dst 14 error closing
   vs dst 14 closed
   vs ssl 14 close
   vs sock 14 free
   vs src 12 closed
   vs ssl 12 close
   vs sock 12 free
```

IP, TCP, and UDP load balancing

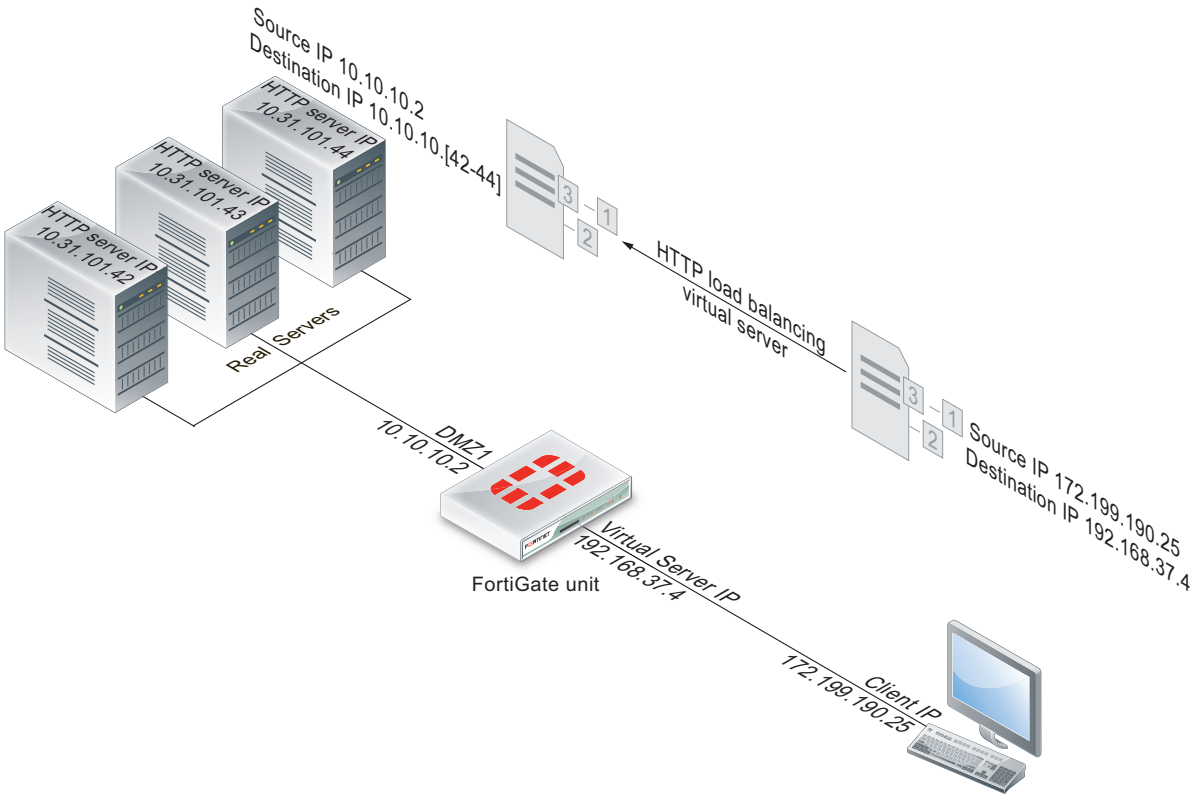
You can load balance all IP, TCP or UDP sessions accepted by the security policy that includes a load balancing virtual server with the type set to IP, TCP, or UDP. Traffic with destination IP and port that matches the virtual server IP and port is load balanced. For these protocol-level load balancing virtual servers you can select a load balance method and add real servers and health checking. However, you can't configure persistence, HTTP multiplexing and SSL offloading.

Example HTTP load balancing to three real web servers

In this example, a virtual web server with IP address 192.168.37.4 on the Internet, is mapped to three real web servers connected to the FortiGate unit dmz1 interface. The real servers have IP addresses 10.10.123.42, 10.10.123.43, and 10.10.123.44. The virtual server uses the **First Alive** load balancing method. The configuration also includes an HTTP health check monitor that includes a URL used by the FortiGate unit for get requests to monitor the health of the real servers.

Connections to the virtual web server at IP address 192.168.37.4 from the Internet are translated and load balanced to the real servers by the FortiGate unit. First alive load balancing directs all sessions to the first real server. The computers on the Internet are unaware of this translation and load balancing and see a single virtual server at IP address 192.168.37.4 rather than the three real servers behind the FortiGate unit.

Virtual server configuration example



GUI configuration

Use the following procedures to configure this load balancing setup from the GUI.

To add an HTTP health check monitor

In this example, the HTTP health check monitor includes the **URL** "/index.html" and the **Matched Phrase** "Fortinet products".

1. Go to **Policy & Objects > Health Check**.
2. Select **Create New**.
3. Add an HTTP health check monitor that sends get requests to http://<real_server_IP_address>/index.html and searches the returned web page for the phrase "Fortinet products".

Name	HTTP_health_chk_1
Type	HTTP
Port	80
URL	/index.html
Matched Content	Fortinet products
Interval	10 seconds
Timeout	2 seconds
Retry	3

4. Select **OK**.

To add the HTTP virtual server and the real servers

1. Go to **Policy & Objects > Virtual Servers**.
2. Select **Create New**.
3. Add an HTTP virtual server that allows users on the Internet to connect to the real servers on the internal network. In this example, the FortiGate wan1 interface is connected to the Internet.

Name	Load_Bal_VS1
Type	HTTP
Interface	wan1
Virtual Server IP	192.168.37.4 The public IP address of the web server. The virtual server IP address is usually a static IP address obtained from your ISP for your web server. This address must be a unique IP address that is not used by another host and cannot be the same as the IP address of the external interface the virtual IP will be using. However, the external IP address must be routed to the selected interface. The virtual IP address and the external IP address can be on different subnets. When you add the virtual IP, the external interface responds to ARP requests for the external IP address.
Virtual Server Port	80

Load Balance Method	First Alive
Persistence	HTTP cookie
Health Check	HTTP_health_chk_1
HTTP Multiplexing	Turn on. The FortiGate unit multiplexes multiple client into a few connections between the FortiGate unit and each real HTTP server. This can improve performance by reducing server overhead associated with establishing multiple connections.
Preserve Client IP	Turn on. The FortiGate unit preserves the IP address of the client in the `X-Forwarded-For` HTTP header.

4. Add three real servers to the virtual server. Each real server must include the IP address of a real server on the internal network.
Configuration for the first real server.

IP Address	10.10.10.42
Port	80
Max Connections	0 Setting **Max Connections** to 0 means the FortiGate unit does not limit the number of connections to the real server. Since the virtual server uses **First Alive** load balancing you may want to limit the number of connections to each real server to limit the traffic received by each server. In this example, the **Max Connections** is initially set to 0 but can be adjusted later if the real servers are getting too much traffic.
Mode	Active

Configuration for the second real server.

IP Address	10.10.10.43
Port	80
Max Connections	0
Mode	Active

Configuration for the third real server.

IP Address	10.10.10.44
Port	80

Max Connections	0
Mode	Active

To add the virtual server to a security policy

Add a wan1 to dmz1 security policy that uses the virtual server so that when users on the Internet attempt to connect to the web server's IP address, packets pass through the FortiGate unit from the wan1 interface to the dmz1 interface. The virtual IP translates the destination address of these packets from the virtual server IP address to the real server IP addresses.

1. Go to **Policy & Objects > IPv4 Policy**.
2. Select **Create New**.
3. Configure the security policy:

Name	Add a name for the policy.
Incoming Interface	wan1
Outgoing Interface	dmz1
Source	all (or a more specific address)
Destination	Load_Bal_VS1
Schedule	always
Service	HTTP
Action	ACCEPT
NAT	Select this option and select **Use Destination Interface Address**.
Log Allowed Traffic	Select to log virtual server traffic

4. Select other security policy options as required.
5. Select **OK**.

CLI configuration

Use the following procedure to configure this load balancing setup from the CLI.

To configure HTTP load balancing

1. Use the following command to add an HTTP health check monitor that sends get requests to http://<real_server_IP_address>/index.html and searches the returned web page for the phrase "Fortinet products".

```
config firewall ldb-monitor
   edit HTTP_health_chk_1
      set type http
      set port 80
      set http-get /index.html
      set http-match "Fortinet products"
      set interval 10
```

```
      set timeout 2
      set retry 3
   end
```

2. Use the following command to add an HTTP virtual server that allows users on the Internet to connect to the real servers on the internal network. In this example, the FortiGate wan1 interface is connected to the Internet.

```
config firewall vip
   edit Load-Bal_VS1
      set type server-load-balance
      set server-type http
      set ldb-method first-alive
      set http-multiplex enable
      set http-ip-header enable
      set extip 192.168.37.4
      set extintf wan1
      set extport 80
      set persistence http-cookie
      set monitor HTTP_health_chk_1
         config realservers
            edit 1
               set ip 10.10.10.42
               set port 80
            next
            edit 2
               set ip 10.10.10.43
               set port 80
            next
            edit 3
               set ip 10.10.10.44
               set port 80
         end
   end
```

3. Use the following command to add a security policy that includes the load balance virtual server as the destination address.

```
config firewall policy
   edit 0
      set name <policy-name>
      set srcintf wan1
      set srcaddr all
      set dstintf dmz1
      set dstaddr Load-Bal_VS1
      set action accept
      set schedule always
      set service ALL
      set nat enable
   end
```

Configure other security policy settings as required.

Example Basic IP load balancing configuration

This example shows how to add a server load balancing virtual IP that load balances all traffic among 3 real servers. In the example the Internet is connected to `port2` and the virtual IP address of the virtual server is 192.168.20.20. The load balancing method is `weighted`. The IP addresses of the real servers are 10.10.10.1, 10.10.10.2, and 10.10.10.3. The weights for the real servers are 1, 2, and 3. The default weight is 1 and does not have to be changed for the first real server.

```
config firewall vip
   edit All_Load_Balance
      set type server-load-balance
      set server-type ip
      set extintf port2
      set extip 192.168.20.20
      set ldb-method weighted
         config realservers
            edit 1
               set ip 10.10.10.1
            next
            edit 2
               set ip 10.10.10.2
               set weight 2
            next
            edit 3
               set ip 10.10.10.3
               set weight 3
            end
   end
```

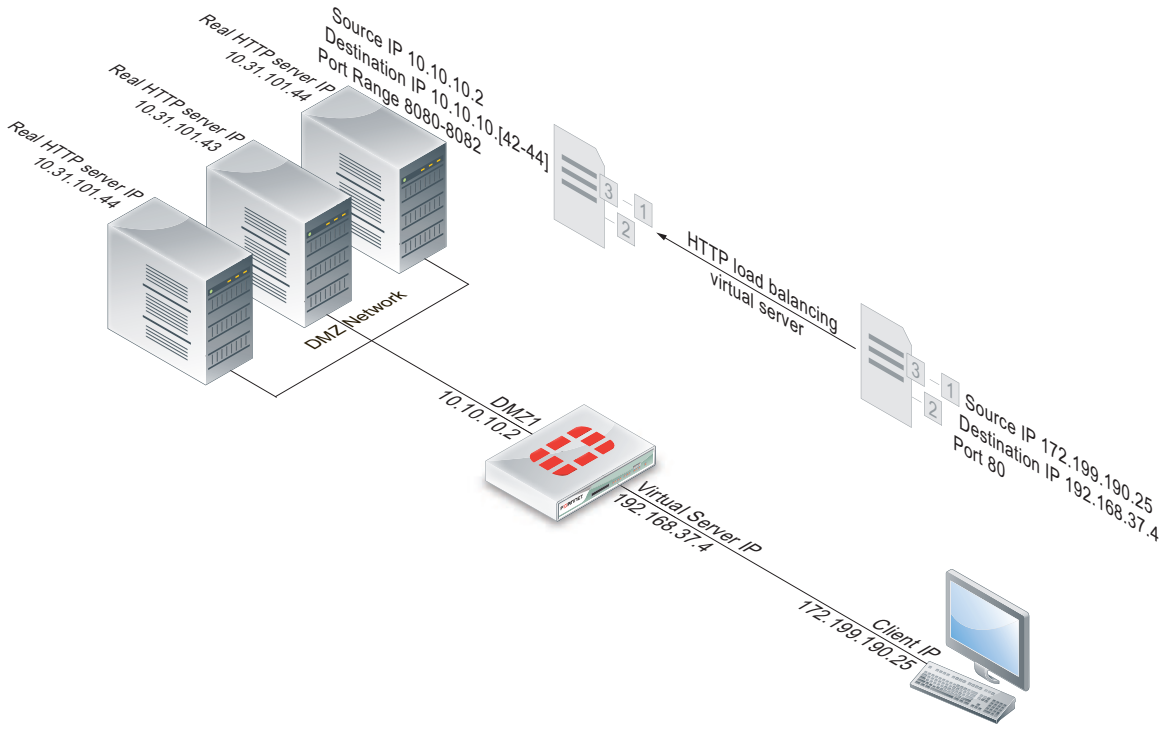
Example Adding a server load balance port forwarding virtual IP

In this example, a virtual web server with IP address 192.168.37.4 on the Internet, is mapped to three real web servers connected to the FortiGate unit dmz1 interface. The real servers have IP addresses 10.10.123.42, 10.10.123.43, and 10.10.123.44. The virtual server uses the **First Alive** load balancing method.

Each real server accepts HTTP connections on a different port number. The first real server accepts connections on port 8080, the second on port 8081, and the third on 8082. The configuration also includes an HTTP health check monitor that includes a URL used by the FortiGate unit for get requests to monitor the health of the real servers.

Connections to the virtual web server at IP address 192.168.37.4 from the Internet are translated and load balanced to the real servers by the FortiGate unit. First alive load balancing directs all sessions to the first real server. The computers on the Internet are unaware of this translation and load balancing and see a single virtual server at IP address 192.168.37.4 rather than the three real servers behind the FortiGate unit.

Server load balance virtual IP port forwarding



To complete this configuration, all of the steps would be the same as in Example Adding a server load balance port forwarding virtual IP on page 54 except for configuring the real servers.

To add the real servers to the virtual server

Use the following steps to add three real servers to the virtual server Load_Bal_VS1. These real servers cause the FortiGate unit to forward HTTP packets to the three real servers on ports 8080, 8081, and 8082.

1. Go to **Policy & Objects > Virtual Servers** and edit the **Load_Bal_VS1** virtual server.
2. Select **Create New**.
3. Add the following three real servers. Each real server must include the IP address of a real server on the internal network and have a different port number.

   Configuration for the first real server.

IP Address	10.10.10.42
Port	8080
Max Connections	0
Mode	Active

   Configuration for the second real server.

IP	10.10.10.43
Port	8081
Max Connections	0
Mode	Active

   Configuration for the third real server.

IP	10.10.10.44
Port	8082
Max Connections	0
Mode	Active

Example Weighted load balancing configuration

This example shows how to using firewall load balancing to load balances all traffic among 3 real servers. In the example the Internet is connected to `port2` and the virtual IP address of the virtual server is 192.168.20.20. The load balancing method is `weighted`. The IP addresses of the real servers are 10.10.10.1, 10.10.10.2, and 10.10.10.3. The weights for the real servers are 1, 2, and 3.

This configuration does not include a health check monitor.

GUI configuration

Use the following procedures to configure this load balancing setup from the FortiGate GUI.

To add the HTTP virtual server

1. Go to **Policy & Objects > Virtual Servers**.
2. Select **Create New**.
3. Add an IP virtual server that allows users on the Internet to connect to the real servers on the internal network. In this example, the FortiGate port2 interface is connected to the Internet.

Name	HTTP_weghted_LB
Type	IP
Interface	port2
Virtual Server IP	192.168.20.20
Load Balance Method	Weighted

All other virtual server settings are not required or cannot be changed.

4. Under **Real Servers** select **Create New**.
5. Add three real servers. Because the **Load Balancing Method** is **Weighted**, each real server includes a weight. Servers with a higher weight receive a more sessions.

Configuration for the first real server.

IP Address	10.10.10.1
Weight	1
Max Connections	0 Setting **Max Connections** to 0 means the FortiGate unit does not limit the number of connections to the real server.
Mode	Active

Configuration for the second real server.

IP Address	10.10.10.2
Weight	2
Max Connections	0
Mode	Active

Configuration for the third real server.

IP Address	10.10.10.3
Weight	3
Max Connections	0
Mode	Active

To add the virtual server to a security policy

Add a port2 to port1 security policy that uses the virtual server so that when users on the Internet attempt to connect to the web server's IP address, packets pass through the FortiGate unit from the wan1 interface to the dmz1 interface. The virtual IP translates the destination address of these packets from the virtual server IP address to the real server IP addresses.

1. Go to **Policy & Objects > IPv4 Policy**.
2. Select **Create New**.
3. Configure the security policy:

Name	Policy name
Incoming Interface	port2
Outgoing Interface	port1
Source	all (or a more specific address)
Destination	HTTP_weghted_LB
Schedule	always
Service	ALL
Action	ACCEPT
NAT	Select this option and select **Use Destination Interface Address**.

4. Select other security policy options as required.
5. Select **OK**.

CLI configuration

Load balancing is configured from the CLI using the `config firewall vip` command and by setting `type` to `server-load-balance`. The default weight is 1 and does not have to be changed for the first real server.

Use the following command to add the virtual server and the three weighted real servers.

```
config firewall vip
   edit HTTP_weghted_LB
      set type server-load-balance
      set server-type ip
      set extintf port2
      set extip 192.168.20.20
      set ldb-method weighted
         config realservers
            edit 1
               set ip 10.10.10.1
            next
            edit 2
               set ip 10.10.10.2
               set weight 2
            next
            edit 3
               set ip 10.10.10.3
               set weight 3
            end
   end
```

Example HTTP and HTTPS persistence configuration

This example shows how to add a virtual server named **HTTP_Load_Balance** that load balances HTTP traffic using port 80 and a second virtual server named **HTTPS_Load_Balance** that load balances HTTPS traffic using port 443. The Internet is connected to port2 and the virtual IP address of the virtual server is 192.168.20.20. Both server load balancing virtual IPs load balance sessions to the same three real servers with IP addresses 10.10.10.2, 10.10.10.2, and 10.10.10.3. The real servers provide HTTP and HTTPS services.

For both virtual servers, persistence is set to **HTTP Cookie** to enable HTTP cookie persistence.

To add the HTTP and HTTPS virtual servers

1. Go to **Policy & Objects > Virtual Servers**.
2. Add the HTTP virtual server that includes HTTP Cookie persistence.

Name	HTTP_Load_Balance
Type	HTTP
Interface	port2
Virtual Server IP	192.168.20.20
Virtual Server Port	80 In this example the virtual server uses port 8080 for HTTP sessions instead of port 80.
Load Balance Method	Static
Persistence	HTTP cookie

3. Under **Real Servers** select **Create New**.
4. Add three real servers.

Configuration for the first real server.

IP Address	10.10.10.1
Port	80
Max Connections	0
Mode	Active

Configuration for the second real server.

IP Address	10.10.10.2
Port	80

Maximum Connections	0
Mode	Active

Configuration for the third real server.

IP Address	10.10.10.3
Port	80
Max Connections	0
Mode	Active

5. Select **OK**.
6. Select **Create New** to add the HTTPS virtual server that also includes HTTP Cookie persistence.

Name	HTTPS_Load_Balance
Type	HTTPS
Interface	port2
Virtual Server IP	192.168.20.20
Virtual Server Port	443
Load Balance Method	Static
Persistence	HTTP cookie

7. Under **Real Servers** select **Create New**
8. Add three real servers.

Configuration for the first real server.

IP Address	10.10.10.1
Port	443
Max Connections	0
Mode	Active

Configuration for the second real server.

IP Address	10.10.10.2
Port	443
Max Connections	0
Mode	Active

Configuration for the third real server.

IP Address	10.10.10.3
Port	443
Max Connections	0
Mode	Active

To add the virtual servers to security policies

Add a port2 to port1 security policy that uses the virtual server so that when users on the Internet attempt to connect to the web server's IP address, packets pass through the FortiGate unit from the wan1 interface to the dmz1 interface. The virtual IP translates the destination address of these packets from the virtual server IP address to the real server IP addresses.

1. Go to **Policy & Objects > IPv4 Policy**.
2. Select **Create New**.
3. Configure the HTTP security policy:

Name	Policy name.
Incoming Interface	port2
Outgoing Interface	port1
Source	all
Destination	HTTP_Load_Balance
Schedule	always
Service	HTTP
Action	ACCEPT
NAT	Select this option and select **Use Destination Interface Address**.

4. Select other security policy options as required.
5. Select **OK**.
6. Select **Create New**.
7. Configure the HTTP security policy:

Name	Policy name.
Incoming Interface	port2
Outgoing Interface	port1
Source	all
Destination	HTTPS_Load_Balance
Schedule	always

Service	HTTPS
Action	ACCEPT
NAT	Select this option and select **Use Destination Interface Address**.

8. Select other security policy options as required.
9. Select **OK**.

CLI configuration: adding persistence for a specific domain

Load balancing is configured from the CLI using the `config firewall vip` command and by setting `type` to `server-load-balance`.

For the CLI configuration, both virtual servers include setting `http-cookie-domain` to `.example.org` because HTTP cookie persistence is just required for the `example.org` domain.

First, the configuration for the HTTP virtual IP:

```
config firewall vip
   edit HTTP_Load_Balance
      set type server-load-balance
      set server-type http
      set extport 8080
      set extintf port2
      set extip 192.168.20.20
      set persistence http-cookie
      set http-cookie-domain .example.org
         config realservers
            edit 1
               set ip 10.10.10.1
            next
            edit 2
               set ip 10.10.10.2
            next
            edit 3
               set ip 10.10.10.3
            end
   end
```

Second, the configuration for the HTTPS virtual IP. In this configuration you don't have to set `extport` to `443` because `extport` is automatically set to `443` when `server-type` is set to `https`.

```
config firewall vip
   edit HTTPS_Load_Balance
      set type server-load-balance
      set server-type https
      set extport 443
      set extintf port2
      set extip 192.168.20.20
      set persistence http-cookie
      set http-cookie-domain .example.org
         config realservers
            edit 1
               set ip 10.10.10.1
```

```
            next
            edit 2
               set ip 10.10.10.2
            next
            edit 3
               set ip 10.10.10.3
            end
      end
```